



Software Quality Assurance

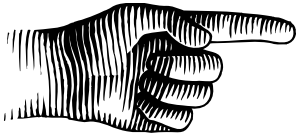
Early and Continuous Throughout the Lifecycle

Justifiable evidence and high confidence that your system performs as expected, when expected, is safe, and is secure.





Outline: Where Are We?



- **Perspective, Challenges, Goals**
- Why Software Quality Assurance
- Problem, Solution, Result
- Software Quality Tools and Life Cycle
- Independent Software Quality Assessment (ISQA)
- Wrap-up
- Glossary





Perspectives Influence Software Quality Goals



Perspectives



DOD

- Warfighter
- Tax Payer



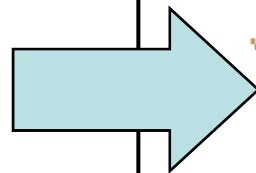
Corporations

- Time To Market
- Reduced Expense
- Increased Profit
- Increased Market Share



Academia

- Payee sets goals
- Theory in practice
- Learning
- Research



Quality Goals

- Safety 
- Security  
- Performance
- Portability
- Reliability  
- Maintainability
- Availability
- Interoperability
- Robust
- Adaptability
- Usability
- Etc.



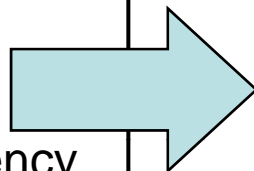
Challenges to Attain Software Quality Goals



Challenges



- Defects
- Politics
- Process
- People
- Money
- Complacency
- Ignorance
- Poor planning
- Data Rights
- Training
- Motivation
- Criteria
- Tools
- Schedule
- SLOC
- Etc.



Quality Goals

- Safety ★
- Security ★★
- Performance
- Portability
- Reliability ★★
- Maintainability
- Availability
- Interoperability
- Robust
- Adaptability
- Usability
- Etc.



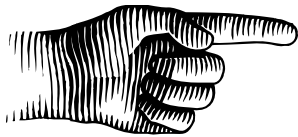
CIO Executive Council™ Poll - 2006



Department of Defense



Outline: Where Are We?



- Perspective, Challenges, Goals
- **Why Software Quality Assurance**
- Problem, Solution, Result
- Software Quality Tools and Life Cycle
- Independent Software Quality Assessment (ISQA)
- Wrap-up
- Glossary





Why Software Quality Assurance?



- Increasing amount & complexity of software-only capabilities
- Growing complexity in COTS, GOTS, and OSS integration
- Example: Service Oriented Architecture (SOA)



F22
1.7M SLOC



**HD DVD
Recorder**
8M SLOC

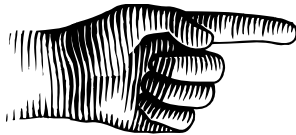
FCS JSF
Ten's of Millions SLOC

Trend: “Hardened” Infrastructure; add more Software!





Outline: Where Are We?



- Perspective, Challenges, Goals
- Why Software Quality Assurance
- **Problem, Solution, Result**
- Software Quality Tools and Life Cycle
- Independent Software Quality Assessment (ISQA)
- Wrap-up
- Glossary





Software Quality Assurance

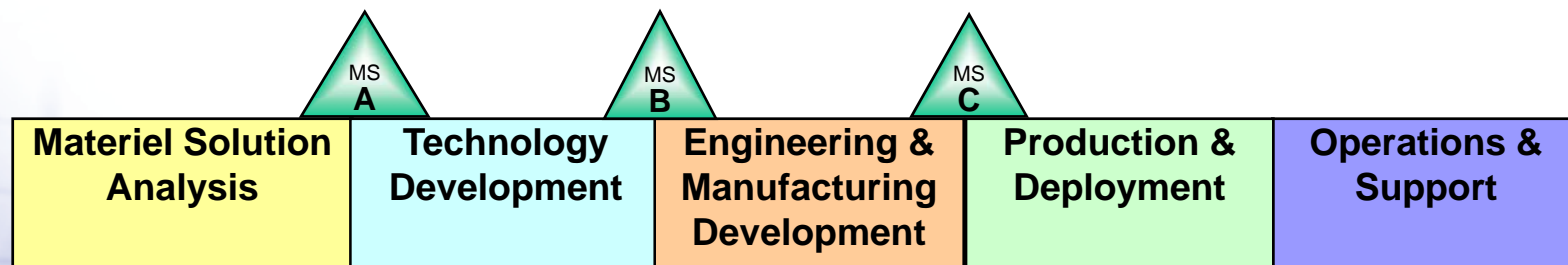
Problem, Solution, Result



Problem: “Software vulnerabilities, malicious code, and software that does not function as promised pose a substantial risk to the Nation’s software-intensive critical infrastructure that provides essential information and services to citizens.” (DHS – Software Assurance in Acquisition: Mitigating Risks to the Enterprise, Oct. 2008)

Solution: Attain justifiable evidence throughout life cycle for your quality goals

Result: Higher confidence that system performs as intended and is not exploitable.





Outline: Where Are We?

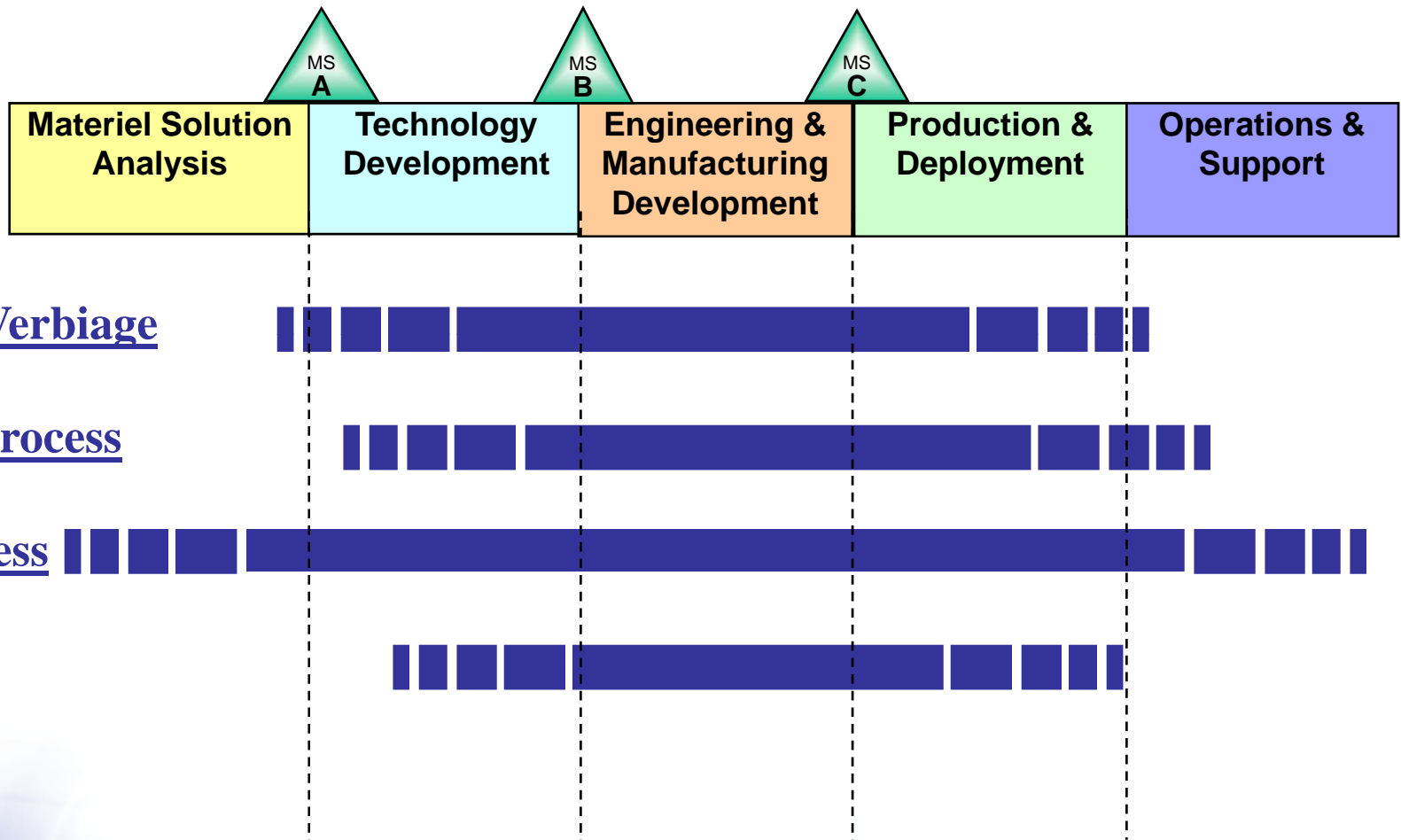
- Perspective, Challenges, Goals
- Why Software Quality Assurance
- Problem, Solution, Result
- **Software Quality Tools and Life Cycle**
- Independent Software Quality Assessment (ISQA)
- Wrap-up
- Glossary





Software Quality Assurance Tools

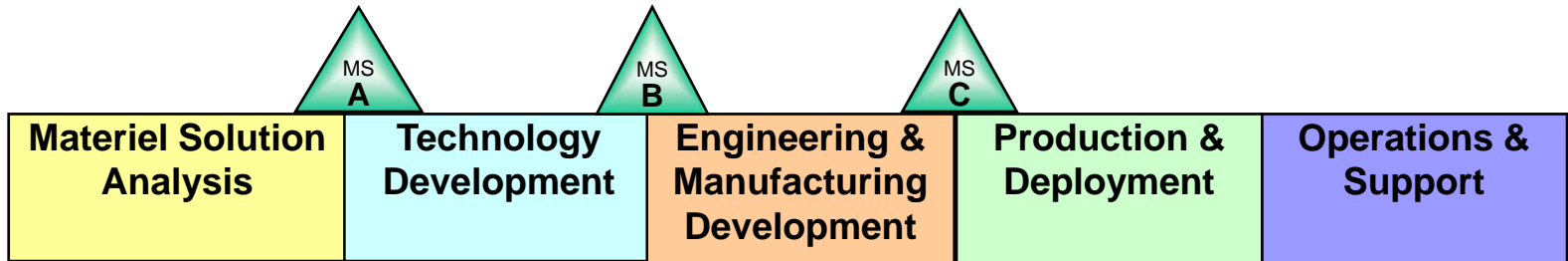
Return on Investment in Life Cycle





Software Quality Assurance Tools

Where to look for “justifiable evidence”!



- Contract Verbiage →**
- Government Data Rights
 - Defects – Forecasted and Actual
 - Visibility at Government’s Discretion
 - Payment Incentives for Defect Reduction
 - Improved Configuration Management
 - Supplier Credentials - clearance, pedigree, etc.
 - Supplier’s evidence of their own assurance claims
 - Independent Software Quality Assessment (iSQA)

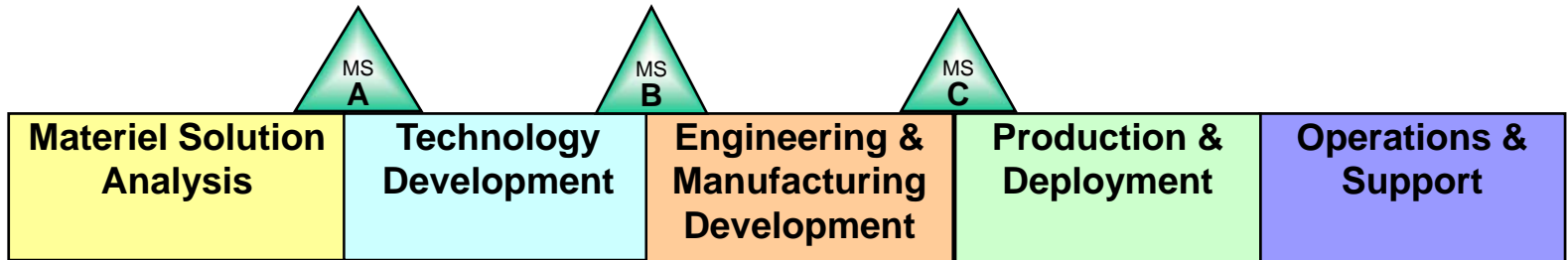
**Ask and You
Shall Receive!**





Software Quality Assurance Tools

Where to look for “justifiable evidence”!



Supplier's Processes → CMMI, ISO, Certifications, etc.

Your Own Processes →

- CMMI, ISO, etc.
- Defense Acquisition Guidebook (Chapter 4, Sys Eng.)
- DoD IA C&A Process (DIACAP) – (DoDI 8510.01)
- “Software Quality” DCSQ-1 (DoDI 8500.2)
- Secure Coding Requirements (IAW DoDD 8500.1)
- Open Source Software Requirements (AR25-2)
- Army Networkiness (AR25-1)
- COTS Security patch process
- Business Best Practices
- Trained Resources

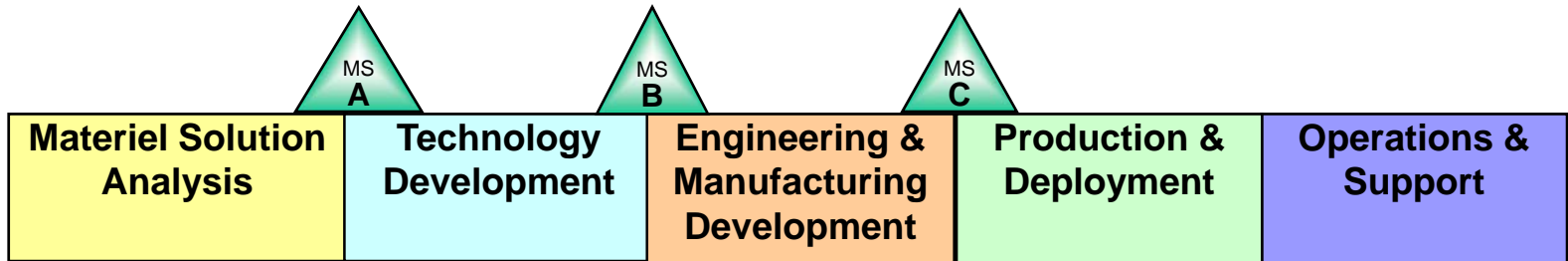
**Look at what is
already available
and required!**





Software Quality Assurance Tools

Where to look for “justifiable evidence”!



Independent Software Quality Assessment (iSQA)

Let the code speak!

- Code-level forensics
- Static and Runtime assessments
- Automated tools reduce time to “find” defects
- Targeted, actionable recommendations to improve
- Subject Matter Experts provide “operational” perspective
- Motivates software developers to do better
- Repeatable measure of software quality



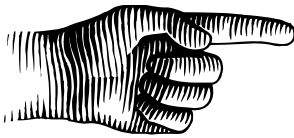
“In general, third-party testing and evaluation provide a significantly greater basis for customer confidence than many other assurance techniques.”
(DHS, *Software Assurance in Acquisition: Mitigating Risks to the Enterprise*, Oct. 2008).





Outline: Where Are We?

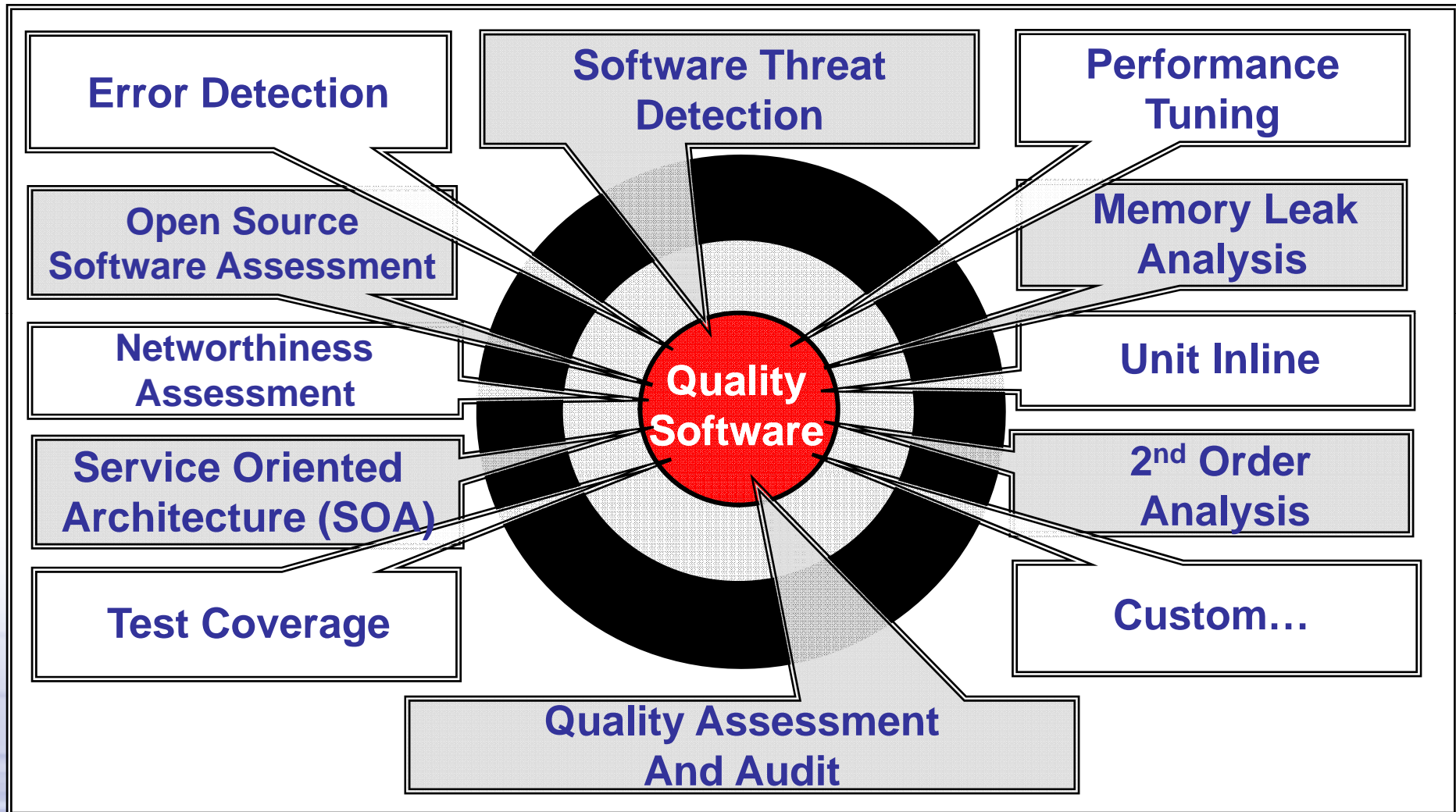
- Perspective, Challenges, Goals
- Why Software Quality Assurance
- Problem, Solution, Result
- Software Quality Tools and Life Cycle
- **ISQA**
- Wrap-up
- Glossary





ISQA Capabilities

Perspectives to let the code speak.






Typical ISQA Customer Profiles



Your profile drives your perspective and ISQA needs.

1 “Code Red” Project 


2 Rapid Prototyping
(Creativity & Speed, not quality) 

3 Legacy System
(Reduce Cost, Reuse
Fresh Coat of Paint) 

4 System Integrator
Syndrome 

5 Conformance
(Industry/Customer
Standards) 

6 Schedule
Compression 

7 Assessment for
Confidence 

8 Security Posture and
Networthiness 



ISQA Return On Investment

Composite Example – 4 Actual Projects

Industry Accepted SW Metrics

\$10,000/bug to Find & Fix a Defect

Finding Bugs = 80% of Cost (\$8,000 per)

	Traditional Defect Cost	"find" \$\$ Avoided	ISQA Cost	Net \$\$ Avoided	ROI
1	335 Defects x \$8,000 =	\$2,680,000	\$545,000	\$2,135,000	4.9
2	219 Defects x \$8,000 =	\$1,608,000	\$219,000	\$1,389,000	7.3
3	1895 Defects x \$8,000 =	\$15,160,000	\$1,214,000	\$13,946,000	12.5
4	70 Defects x \$8,000 =	\$560,000	\$140,000	\$420,000	4.0
	2519 Defects x \$8,000 =	\$20,008,000	\$2,118,000	\$17,890,000	9.5



ISQA Artifacts



What Justifiable Evidence Should You Expect?

▪ Scorecard Summary

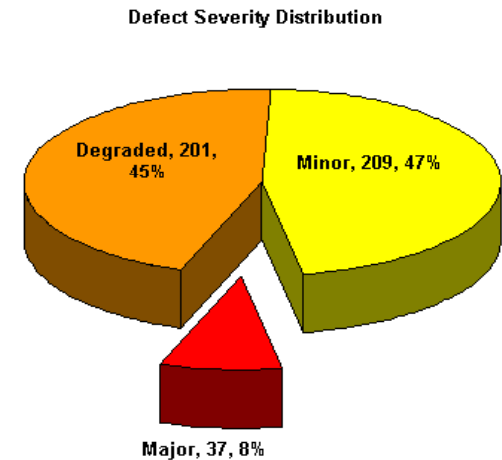
- Quick assimilation of data (e.g. graphics)
- Highlight areas for improvement
- Management / Executive audience

▪ Detailed Technical Report

- Description of findings
- Qualitative description of coverage
- Short, Medium, and Long Term actionable recommendations

▪ Raw Data – per defect

- Module, LOC, severity, problem, actionable recommendation
- Formatted for ease of use (e.g. Common separated values, Excel spreadsheet, links from defect to actual line of code, etc.)





Scorecard Example



Against DISA Application Security and Development STIG

CODE INSPECTION RESULTS									
		Instances	CAT I	CAT II	Minor	Bad Style	No Defect	Informational	% Assessed
INSPECTION ATTRIBUTES									
APP No.									
3050	Defects: Dead or Dormant Code	388	0	2					
3100	Defects: Apparent Unclosed Stream	10	0	2					
3120	Exception Handling Attributes: Error Handling	2353	0	25					
3120	Exception Handling Attributes: The program can potentially dereference a null pointer, thereby causing a segmentation fault.	2300	0	115					
3100	Defects: Unreleased Resource	225	0	0					
2060.4	Defects: Dangerous Functions	10	0	0					
3120	Exception Handling Attributes: Leaving Return Value Of Symbol	4	0	0					
DATA SECURITY									
3150.2	Cryptography: Standard pseudo-random number generators cannot withstand cryptographic attacks	19	0						
3310	Password Management: Credential Management- Passwords Stored as Clear Text	6	2	0					100
INPUT VALIDATION									
3570	Command Injection: Executing commands that include un-validated user input can cause an application to act on behalf of an attacker.	5	0	1	0	0	4	0	100
3510	General Input Validation: No Usable Struts Validation	490	0	15	0	0	30	3	10
3540.1	SQL Injection: SQL Injection User Input	583	0	0	0	0	21	562	100
3580	Cross Site Scripting: CrossSiteScripting	110	0	0	0	0	0	109	99
3530	General Input Validation: Web Character Set	382	0	0	0	0	0	382	100
3520	General Input Validation: Trust Boundary Violation	125	0	0	0	0	0	3	2
3540.1	SQL Injection: SQL Injection User File	316	0	0	0	0	18	298	100
PORTABILITY AND SECURITY									
3600	Code Hacking Attributes: Canonical Representation Vulnerabilities	79	0	0	0	0	4	75	100
3630.3	Code Hacking Attributes: Deprecated Thread Functions	600	0	0	0	0	0	600	100
SUMMARY OF ISSUES FOUND			2	160	0	0	1626	4775	
KEY DEFECTS			162						
ALL DEFECTS			162						

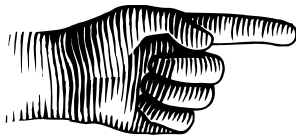
- Category of Finding
- STIG Requirement Number
- Validate "Real and Actionable"
- Actionable Results Feed Into developer's "Get Well Plan" for the system.





Outline: Where Are We?

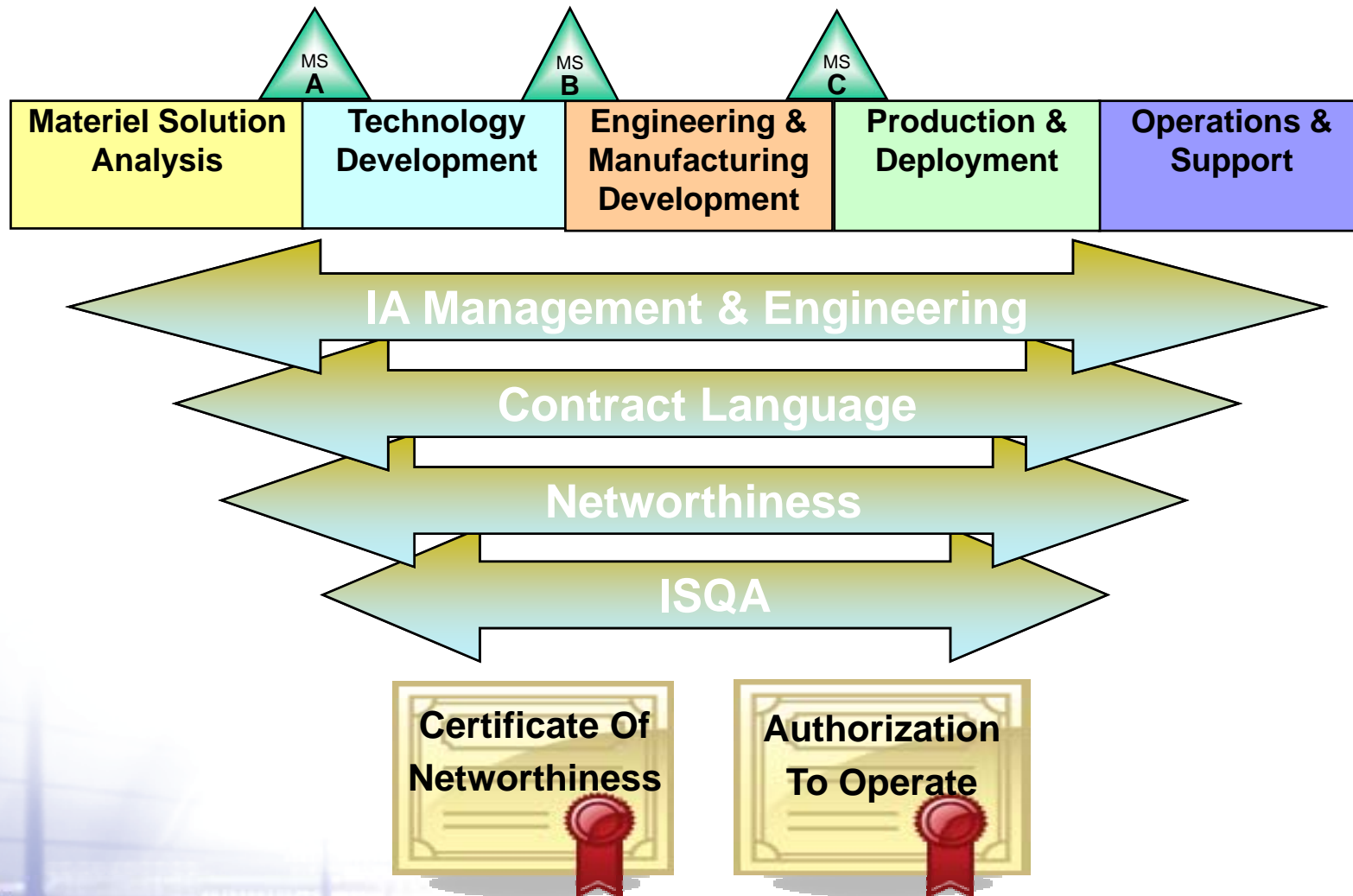
- Perspective, Challenges, Goals
- Why Software Quality Assurance
- Problem, Solution, Result
- Software Quality Tools and Life Cycle
- Independent Software Quality Assessment (ISQA)
- **Wrap-up**
- Glossary





DoD / Army Software Quality Assurance

Life Cycle Evidence for Confidence to Operate





Glossary



- **AR** – Army Regulation (e.g. AR25-2)
- **Assurance** - a statement or indication that inspires confidence, a guarantee
- **C&A** – Certification and Accreditation
- **CON** – Certificate of Networthiness for the Army
- **COTS** – Commercial Off the Shelf software
- **DHS** – Department of Homeland Securities
- **DIACAP** – Defense Information Assurance Certification and Accreditation Process
- **DISA** - Defense Information Systems Agency
- **DoDD** – Department of Defense Directive
- **DoDI** – Department of Defense Implementation
- **GOTS** – Government Off the Shelf software
- **Life Cycle** – all phases of a system’s life from concept through disposal
- **OSS** – Open Source Software
- **Quality** – an essential or distinctive characteristic, property, or attribute
- **Software Assurance** - “...the level of confidence that software is free from vulnerabilities, either intentionally designed into the software or accidentally inserted at any time during its life-cycle, and that it functions in the intended manner.” [CNSSI no 40090]
- **STIG** – Security Technical Implementation Guide





Presenter's Credentials and Contact Information





About The Presenter

Credentials



Name: Bruce Weimer

Employer: US Army – CECOM LCMC Software Engineering Center,
Software Assurance Division

Experience:

- 4+ years in Civilian Army – System's Engineer
- 23 years in Industry – Pharma., Financial, Telecom, SW products
- Full Software Life-cycle Development
- Software Quality Assurance
- Information Assurance (IA)
- Process Improvement – CMMI, Lean Six Sigma, ISO
- Content/Document Management
- Workflow and Process Improvement
- Masters in Software Engineering

Contact: bruce.weimer@conus.army.mil, 732.532.5020 / DSN 992

